МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

Допустить к защите

Заведующий кафедрой,

д-р техн. наук, проф.,

\_\_\_\_\_\_\_\_\_ А.А. Еремин

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**(БАКАЛАВРСКАЯ РАБОТА)**

**РАЗРАБОТКА СИМУЛЯТОРА АВТОПИЛОТА ТРАМВАЯ**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Д.Е. Гиренко

(подпись)

Направление подготовки 02.03.02 — «Фундаментальная информатика и информационные технологии»

Направленность (профиль) «Вычислительные технологии»

Научный руководитель,

канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Т.А. Приходько

(подпись)

Нормоконтролер,

канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е.А. Нигодин

(подпись)

Краснодар

2024

**РЕФЕРАТ**

Выпускная квалификационная работа 53 стр., 9 ч., 58 рис., 16 формул, 12 источников, 4 приложения.

АВТОПИЛОТ, YOLO, НЕЙРОННЫЕ СЕТИ, СИМУЛЯТОРЫ, РАБОТА С ПАМЯТЬЮ ПРОЦЕССА, ТРАМВАЙ

Объектом исследования являются архитектура и особенности разработки автопилота трамвая для симулятора.

Предметом исследования в данной работе являются функциональность автопилота, его автономность и эффективность для симуляции, а также реализация оптимальных методов общения автопилота и симулятора.

Цель работы: разработка программы автопилота для управления трамваем в симуляторе.

Методологическая основа включает в себя анализ литературы и документации, сравнительный анализ различных подходов и практик, математическое и программное моделирование, экспериментальное тестирование различных методов и практик в разработке автопилотов, формулирование выводов.

Научная новизна работы заключается в изучении возможности использования автопилота для управления трамваем в симуляторе, а также выкладывание результатов исследования в общий доступ.

В результате решения поставленных задач была реализована программа автопилота для симулятора управления трамваем TramSim Vienna, использующая YOVLv8 для детекции объекты с видеопотока и получающая значения скорости трамвая напрямую с памяти процесса, которая способна проходить маршрут трамвая без столкновений, производить посадку-высадку пассажиров на каждой остановке и следовать сигналам светофоров.

**СОДЕРЖАНИЕ**

[Введение 5](#_Toc104743664)

[1 Формальная постановка задачи определения профориентации 7](#_Toc104743665)

[2 Теоретические основы Data Mining текстовых данных 8](#_Toc104743666)

[2.1 Понятие Data Mining 8](#_Toc104743667)

[2.2 Методы классификации тональности 8](#_Toc104743668)

[2.2.1 Методы, основанные на правилах и словарях 8](#_Toc104743669)

[2.2.2 Машинное обучение с учителем 9](#_Toc104743670)

[3 Теоретические основы нечеткого анализа 9](#_Toc104743671)

[3.1 Понятие лингвистической переменной и терма 9](#_Toc104743672)

[3.2 Построение функции принадлежности косвенным методом опроса Т. Саати 10](#_Toc104743673)

[3.3 Построение функции принадлежности косвенным методом опроса А.П. Ротштейна 11](#_Toc104743674)

[4 Рассматриваемая классификация профессий 14](#_Toc104743675)

[5 Разработка социальной сети 17](#_Toc104743676)

[5.1 Предпосылки разработки социальной сети 17](#_Toc104743677)

[5.2 Программная реализация серверной части социальной сети 17](#_Toc104743678)

[5.3 Программная реализация клиентской части социальной сети 18](#_Toc104743679)

[6 Разработка системы Data Mining 21](#_Toc104743680)

[6.1 Инструменты анализа данных 21](#_Toc104743681)

[6.2 Набор данных для обучения моделей 21](#_Toc104743682)

[6.2.1 Набор данных для обучения моделей классификации тональности текста 21](#_Toc104743683)

[6.2.2 Набор данных для обучения моделей классификации тем документов 22](#_Toc104743684)

[6.3 Разработка системы классификации тональности текста 23](#_Toc104743685)

[6.4 Разработка системы классификации тем документов 34](#_Toc104743686)

[7 Разработка экспертной системы 38](#_Toc104743687)

[7.1 Построение экспертной системы 38](#_Toc104743688)

[7.2 Сбор оценок экспертов 39](#_Toc104743689)

[7.3 Реализация и тестирование экспертной системы 39](#_Toc104743690)

[8 Схема работы системы прогнозирования 43](#_Toc104743691)

[9 Тестирование системы прогнозирования профориентации 44](#_Toc104743692)

[Заключение 51](#_Toc104743693)

[Список использованных источников 52](#_Toc104743694)

# **ВВЕДЕНИЕ**

В современном мире технологические достижения стремительно преобразуют нашу жизнь. Существенный прогресс в области нейросетей открывает перед нами возможности создания сложных алгоритмов, способных автоматизировать процессы управления транспортными средствами, включая трамваи.

Трамваи играют значительную роль в системе общественного транспорта России, обеспечивая массовый перевоз пассажиров в крупных городах. Их эффективное и безопасное функционирование не только важно для плавного движения городской жизни, но и неразрывно связано с общественным комфортом и экономическим развитием регионов. Вместе с тем, несмотря на значимость трамвайного транспорта, существует потребность в постоянном движении вперёд, освобождая людей от рутинной работы и повышая уровень безопасности и эффективности транспортной системы. В этом контексте разработка автопилота для трамваев выступает как важное направление инноваций, направленных на улучшение жизни городского населения и оптимизацию транспортных процессов.

Целью данной работы является изучение и реализация подходов написания автопилота для симулятора управления трамваем для того, чтобы на практике оценить весь объём предметной области и столкнуться с любыми возможными проблемами разработки, чтобы решить.

Данная работа состоит из 7 глав.

Первая глава содержит теоретические сведения касательно автопилотов и известные на данный момент знания о программном управлении трамваем.

Вторая глава содержит описание выбранного для работы симулятора.

Третья глава содержит анализ и методы решения проблемы подключения потоков входной информации и выходных сигналов управления.

В четвертой главе рассматривается вся работа с нейронной сетью, решающей задачу детекции объектов на изображении.

В пятой главе описывается разработка модуля принятия решения.

В шестой главе описывается итоговая система, объединяющая все реализованные решения.

В седьмой главе проводится тестирование разработанной системы на маршруте в симуляторе.

# **1 Общая теория автопилотов**

## **1.1 Определение и использование автопилота в транспорте**

Автопилот — программно-аппаратный комплекс, управляющий транспортным средством, для следования определённому маршруту или выполнения особых задач. То есть система автопилотирования – это не только программный код, вычисляющий правильную последовательность действий для транспорта, но и ещё весь набор датчиков, позволяющих получать необходимую информацию, и устройства вывода, передающие цифровые сигналы в механические части транспорта. Среди автопилотов наземного транспорта, зачастую, такой комплекс устанавливается в уже готовый транспорт как дополнение в виде камер, спутникового приёмника, лидара, радара, компьютера и проводов, соединяющих всё.

Стоит ещё отметить, что не все транспортные средства способны подключиться к автопилоту, а только те, которые обладают CAN-шиной — часть системы управления автомобилем. Система использует двоичные коды, чтобы обмениваться информацией между различными частями транспортного средства. Весь процесс протекает в реальном времени, без задержек, а также отличается высочайшей надёжностью. В 2018 году использование CAN-шины в разъёме для диагностики стало обязательным условием для автомобилей, которые продаются в США, так что использование этой шины не только в личном транспорте, но и в любой технике уже давно является стандартом.

Большинство людей знают об автопилотах из авиации, где они используются повсеместно. По больше части причина этому то, что в воздушном пространстве самолёту достаточно выполнять вполне простые задачи: лететь по заданному маршруту и держать судно в правильно положении. С этим могут справиться даже обычные программы с математической логикой. Однако, в наземном транспорте всё становится сложнее в виду множества различных объектов, которые могут встретиться на дороге и на которые необходимо правильно реагировать, в том числе и другие участники дорожного движения. Причём, чем меньше масштаб, тем большая точность требуется от управляющего комплекса. Если для большегрузного автомобиля с задачей доставки груза через несколько километров по территории закрытого склада ещё можно воспользоваться только навигацией с GPS сигнала, то для параллельной парковки личного автомобиля компьютеру понадобится точное понимание расстояния до объектов, для чего используют радар или даже лидар.

## **1.2 Определение автопилота по уровням**

Как часто бывает в технике для того, чтобы достичь задуманного решения необходимо наметить путь совершенствования решения и решать задачи последовательно. Аналогично и среди автопилотов выделяют пять уровней:

1. Водительская помощь. На уровне 1 присутствует частичная автоматизация, при которой система может управлять либо ускорением и торможением, либо рулевым управлением, но не обеими функциями одновременно. Примером может служить адаптивный круиз-контроль, который поддерживает заданную скорость и дистанцию до впереди идущего автомобиля, требуя от водителя управления рулем;
2. Частичная автоматизация. Предполагает наличие системы, которая способна одновременно управлять и ускорением/торможением, и рулевым управлением. При этом водитель обязан постоянно контролировать дорогу и быть готовым взять управление в любой момент;
3. Условная автоматизация. Система управления транспортным средством способна самостоятельно выполнять все функции вождения в определенных условиях и средах, таких как шоссе. Водитель может не следить за дорогой, но должен быть готов вмешаться по запросу системы;
4. Высокая автоматизация. Система способна самостоятельно управлять транспортом в большинстве ситуаций без вмешательства водителя. Водитель может полностью отвлечься от управления в течение всего времени движения в пределах определенных зон или условий, например, в городских районах с низкой скоростью или на определенных маршрутах. Если система сталкивается с условиями, которые она не может обработать, она может безопасно остановить транспорт;
5. Полная автоматизация. Транспорт способен выполнять все задачи по управлению транспортным средством в любых условиях и на любых дорогах без какого-либо вмешательства со стороны человека. Водитель становится пассажиром и не требуется вообще для управления автомобилем. На данном уровне могут отсутствовать даже традиционные элементы управления, такие как педали и рулевое колесо.

Более наглядный примера можно посмотреть на рисунок 1.



Рисунок 1 – Уровни автономности.

Так что сейчас в автопилотировании стоит вопрос о том, как создать автопилот, соблюдающий все правила дорожного движения и способного разумно реагировать на любые непредвиденные ситуации в пути, подобно человеку. Впрочем, в последние годы отличные результаты показывают автопилот от компании Tesla, использующий только камеры для управления автомобилем, а также автопилотные такси от российской компании Яндекс. Однако их решение нельзя назвать полным автопилотом в виду технических ограничений. Кроме того, ни одна из компаний не стремится к созданию полного автопилота для дорог общего пользования в виду того, что ни одно законодательство мира не спешит определять законы для транспортного средства, которое работает на автопилоте. Для решения юридического вопроса в той же России необходимо будет много времени и сил со стороны компаний-производителей автопилотов. Но помимо этой преграды, автопилотам ещё придётся пройти через недоверие людей, которые не захотят доверять свою жизнь машине.

## **1.3 Особенности разработки автопилота для трамвая**

В некоторых странах, помимо самолётов, ещё и поезда используют автопилот или автоматизированные системы управления, так как задач, которые должен выполнять водитель, например, грузового поезда, намного меньше. Поэтому трамвай — это идеальная «песочница» для полного автопилота автомобиля, так как он совмещает в себе, как и облегчения от вождения поезда, так и должен соблюдать правила дорожного движения и участвовать во взаимодействии с пешеходами и автомобилями.

Трамвайный маршрут всегда известен, транспортное средство не может с него сойти. Кроме того, маршрут, зачастую, регулярный, а значит можно проехать заранее и разметить точки со светофорами и прочим, чтобы системе было легче их распознавать. Трамвай не может перестроиться из полосы в полосу, что является огромной нагрузкой автопилота автомобиля, а здесь же и вовсе отсутствуют повороты. Кроме того, тормоз за счёт железнодорожного полотна работает очень резко и способен остановить трамвай практически мгновенно. А с другой же стороны, для трамвая добавляется задача особого поведения на остановках, чтобы производить посадку и высадку пассажиров.

Благодаря всем особенностям управления, трамвай имеет потенциал стать первым общественным транспортом, который сможет полностью перейти на автоматическое пилотирование или хотя бы заменить водителя на бортинженера, что значительно уменьшит кадровый голод этой профессии. В данный момент реализацией этой идеи занимается дочерняя компания Сбербанка – Cognitive Pilot. У них уже имеется успешный проект по написанию автопилота для сельскохозяйственных комбайнов, знания из которого они и применяют сейчас экспериментальных автопилотов, которые уже тестируются на дорогах Петербурга. За пару лет жизни проекта компания смогла представить статистику, доказывающую эффективность использования автопилота перед ручным управлением. За счёт строго выполнения правил и машинной точности удалось значительно увеличить безопасность поездки для пассажиров и уменьшить износ техники и железнодорожных путей, несмотря на неприязнь к новой технологии водительского состава.

Именно этот проект вдохновил меня на создание подобного автопилота для управления трамваем. В виду того, что ни одно депо не выделило бы мне для написания работы реальный трамвай, было решено использовать симуляцию этого транспортного средства.

# **2 Описание симулятора TramSim**

Задача написания автопилота для реального транспортного средства совсем не тривиальная задач в виду сложности моделирования всех возможных физических процессов и событий в компьютерной среде. В виду ограниченности ресурсов, было решено использовать симулятор трамвая. Он, конечно, создаёт намного меньше реальных проблем, с которыми сталкиваются команды разработки на реальной технике, но, с другой стороны, симулятор и сильно ограничен по своим программным возможностям, например, полным отсутствием GPS, удобной CAN-шины и многим другим. Так что предстояло решать сходные проблемы, но в цифровой среде, с расчётом на то, что созданная мной система может быть использована в будущем на реальном транспорте при минимальных изменениях модулей получения информации и передачи выходного сигнала.

Для создания симулятора работы трамвая в городском окружении было решено использовать компьютерную игру TramSim Vienna [2]. Основным критерием выбора стала разработка на игровом движке Unreal Engine, что обеспечивает высокое качество графики и приближает визуальное восприятие к реальности. Этот выбор позволяет пользователям получить максимально реалистичный опыт управления трамваем, воспроизводящий условия работы настоящего транспортного средства в городской среде.

Одним из важных преимуществ TramSim Vienna является опыт разработчиков в создании симуляторов различных транспортных средств. Их аккуратность и внимание к деталям отражаются в реализации транспортного средства в игре, а также в его отзывчивости к действиям игрока. Разработчики серии игр TramSim обладают значительным опытом, который позволяет им создавать симуляторы, максимально приближенные к реальной работе транспортных средств, что делает выбор TramSim Vienna логичным и обоснованным для целей данного исследования. Пример графики можно увидеть на рисунке 2.



Рисунок 2 – Пример графики из TramSim Vienna.

В данной игре можно полностью проделать основную работу водителя трамвая на определённом маршруте, а именно следовать ему согласно правилам дорожного движения, правильно реагировать на сигналы светофоров и производить посадку и высадку пассажиров на остановках. В симуляторе присутствуют машины, имитирующие реальный трафик, а также пешеходы, которые могут приходить на остановку и перемещаться на трамвае.

# **3 Налаживание потоков ввода и вывода**

## **3.1 Установка камеры и захват изображения**

В контексте реальных систем сбора информации о внешнем мире, обычно используются разнообразные датчики, такие как камеры, радары, лидары, а также информация с датчиков, подключенных к CAN-шине трамвая. Однако, в случае с симулятором TramSim Vienna, ограничения на доступ к исходному коду не позволяют модифицировать транспортное средство или добавить новые сенсоры для сбора данных. Из-за этого, единственный способ получения информации о находящихся вокруг трамвая объектов являются виды: от первого лица, из кабины, свободная камера в системе координат трамвая и полностью свободная камера. Переключение между ними осуществляется при помощи горячих клавиш, причём мгновенно и с сохранением положения, что крайне удобно.

Вид от первого лица водителя не подходит, так как большая часть обзора закрыта приборной панелью и кабиной трамвая. Пример такого кадра можно увидеть на рисунке 3. Чтобы не терять информацию, было решено использовать свободную камеру, которая следует за трамваем, установив её на носу транспорта. Для того, чтобы при каждом запуске камера занимала одно и то же место, был написан специальный скрипт, эмулирующий движения курсором. Остальные же виды было решено использовать только в случае необходимости.



Рисунок 3 – Вид из кабины.

Теперь, когда основная камера находится на своём месте необходимо получить с неё картинку в программу автопилота. Для получения потока видео из игры, можно было воспользоваться специальными программами записи, но для оптимальности было решено воспользоваться готовым API от Windows, которым можно было воспользоваться в одноимённой библиотеке Python-а «windows\_capture». Достаточно было указать имя открывающегося приложения.

## **3.2 Получение скорости и ускорения**

В реальных системах автопилота получение скорости и многих других данных – тривиальная задача. Достаточно подключить компьютер к CAN-шине и начать читать поток. Однако внутри симулятора такой метод решения задачи недоступен.

Первоначально были предприняты попытки с помощью игровой консоли выводить на экран число, а после его считывать вместе со всей картинкой. Однако, все попытки найти подобную команду, даже прямое письмо разработчикам игры с просьбой помочь, не увенчались успехом. Тогда появилась идея о том, чтобы использовать возможность быстрого переключения между камерой из кабины водителя и свободной камеры на носу трамвая для того, чтобы разделить все считываемые кадры игры на кадры с изображением окружающей местности впереди и на кадры с изображением спидометра. Для перевода кадра, содержащего спидометр, в вещественное число скорости трамвая использовалась функция, что из всего кадра вырезает отдельный прямоугольник со спидометром, после чего подсчитывает количество пикселей зелёного (в цвет полоски спидометра) цвета. Однако этот метод имел крайне большую ошибку и не мог использоваться для стабильного использования. Чтобы добиться меньшей ошибки, пришлось бы использовать ещё одну нейросеть, определяющую по картинке спидометра, какую скорость он показывает, или же создать ещё больше эвристик для подсчёта пикселей.

Тогда было решено, по аналогии с CAN-шиной, получить доступ к памяти процесса игры на прямую и оттуда начать получать значение скорости движения трамвая. Для работы с памятью процесса и получения нужного адреса значения было решено воспользоваться Cheat Engine.

Cheat Engine — это программа с открытым исходным кодом, которая позволяет пользователям изменять параметры в компьютерных играх. Она обеспечивает возможность изменения числовых значений. Пользователи могут сканировать память игры, находить адреса и значения, которые могут быть изменены, и создавать чит-таблицы для использования в игре. В основном программу применяют для редактирования числовых значений в видеоиграх для получения каких-либо нечестных бонусов. Аналогом Cheat Engine является и Art Money, однако за счёт долголетия жизни проекта в качестве программы с открытым исходным кодом, функционал Cheat Engine опережает Art Money.

Примером такого расширенного функционала являются инструменты для автоматизации определенных действий в игре через запись и воспроизведение скриптов, а также получение информации о постоянных указателях на те или иные значения в памяти. Именно этими функциями и предстояло воспользоваться, чтобы прямо с процесса работы симулятора получать информацию о скорости.

Операция получения постоянного адреса скорости трамвая с помощью Cheat Engine состояла из нескольких шагов:

1. Из всего множества адресов памяти, которые использует процесс «TramSimVienna-Win64-Shipping.exe» выделить тот, который хранит в себе значение скорости трамвая. Адрес находился достаточно тривиально – после ускорения трамвая в игре адреса отсеивались по признаку увеличения значения, хранимого по адресу, после трамвай замедлялся и оставались лишь значения, что уменьшились с последнего отсеивания;
2. Найти все указатели процесса, что указывали на этот адрес;
3. Перезагрузить игру, тем самым, обновив всю память и поменяв адресы. Выполнить первый пункт и среди множества всех указателей с прошлого пункта оставить только те, которые сейчас указывают на новый адрес значения скорости;
4. Выполнять третий пункт до тех пор, пока не будет уверенности в хотя бы одном указателе, который при любых запусках игры всегда указывает на адрес, в котором лежит значение скорости.

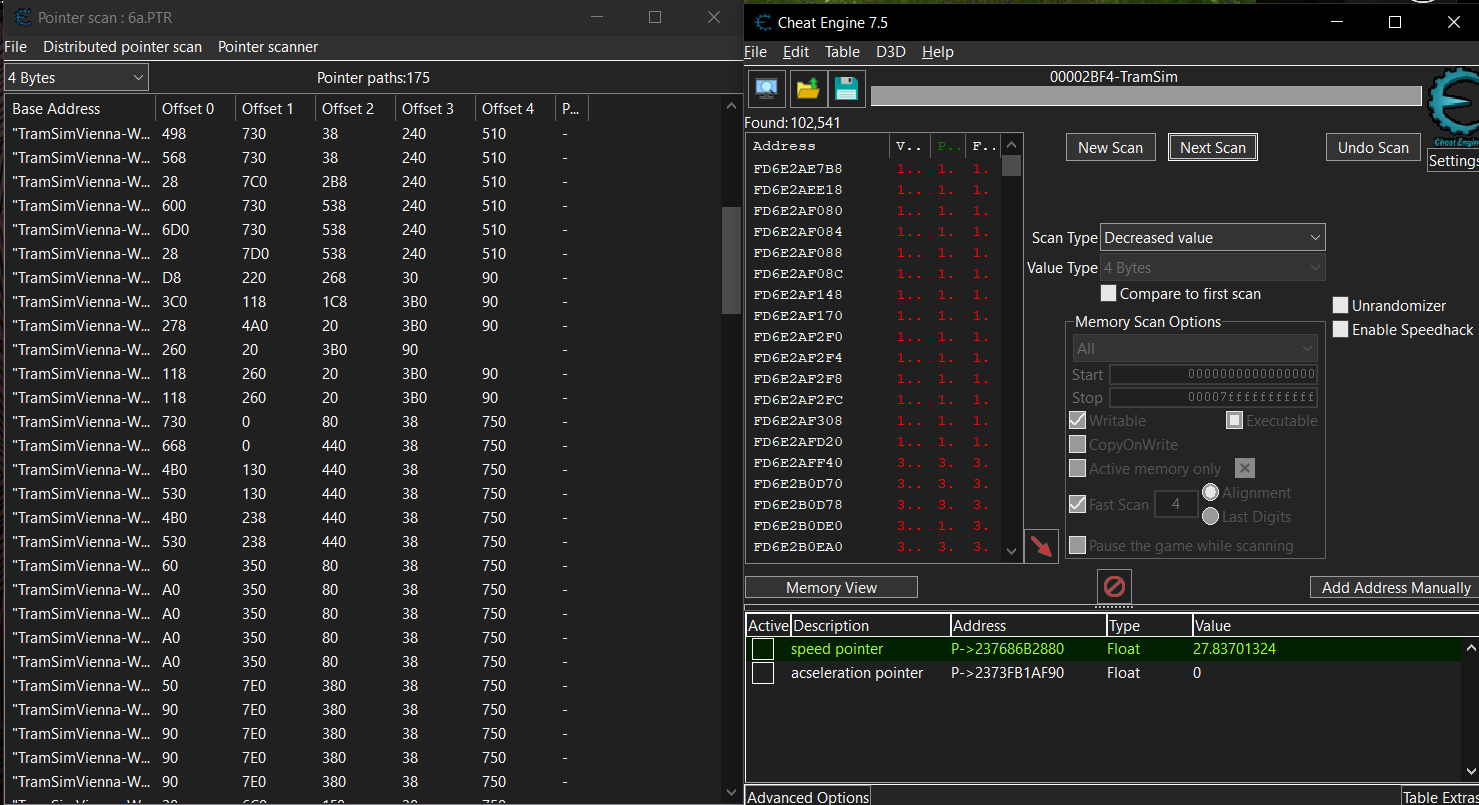


Рисунок 4 – Пример работы Cheat Engine.

Выделив нужный указатель, записываем его данные в программу автопилота, которая, буквально, получает доступ к памяти процесса и читает хранящееся по адресу значение. Эта операция происходит моментально, так как это самая стандартная операция чтения памяти. Благодаря этому автопилот, подобно физической своей версии, способен прямо из процесса работы трамвая получать его скорость и дальше использовать при принятии решений.

Аналогично скорости, было настроено получение из симулятора ускорения трамвая, для более лёгкого его управления.

## **3.3 Программное управление трамваем**

Управление транспортным средством в TramSim Vienna осуществляется при помощи клавиатуры и мыши. Так что для передачи сигналов управления трамваю достаточно эмулировать их, с чем может справиться любой язык программирования. Так как симулятор достаточно отзывчив на короткие сигналы управления в виду того, что игрок управляет ускорением двигателя, было решено не использовать параллельный процесс для управления, а просто занимать некоторое время основного на нажатие клавиши.

# **4 Настройка нейронной сети для решения задачи детекции**

## **4.1 Описание объектов, необходимых для детекции**

Один из упомянутых плюсов писать в симуляции – все возможные ситуации на дороге возможно описать. Переход людей через пешехода, движение впередистоящего автомобиля, светофоры и подобные вещи – то немногое, с чем придётся столкнуться модулю принятия решений. Никаких животных, вышедших на рельсы или неожиданной аварии на дорожных путях. Конечно, в планах писать автопилот с расчётом на то, что он может использоваться и на реальном транспорте, но для обработки таких «форс мажоров» требуется конкретные примеры как изображений таких ситуаций, так и действий водителя, что невозможно получить, используя симуляцию.

Главная задача трамвая – безопасно пройтись по маршруту, останавливаясь на каждой остановке согласно расписанию. Свойство безопасности достигается, по большей части, благодаря следованию сигналам светофоров и резким остановкам в случае, если трамвай ожидает столкновение с каким-либо объектом. То есть сразу для реализации этого свойства нам необходимо считывать с изображения людей, машин и трамвайных светофоров.

Для того, чтобы пройтись по остановкам, необходимо эти самые остановки воспринимать. Для реальных трамваем используются GPS датчики и другие решения от постоянного маршрута. В игре же только при помощи камеры придётся определять места, где трамвай должен остановиться. Благо, это вполне возможно благодаря жёлтым кружкам между рельсами, которые обозначают конкретную точку, где должен трамвай остановиться. Так что точно в объекты детекции необходимо добавить маркеры остановки. Пример изображения такого маркера можно увидеть на рисунке 5.



Рисунок 5 – Маркер остановки.

Таким образом, с помощью перечисленного набора у автопилота получится передвигаться по городу, не нарушая правила дорожного движения, определять трамвайные остановки, избегать случайных столкновений с людьми или машинами, что вполне будет достаточно для автономности.

## **4.2 Обоснование выбора YOLOv8**

Детекция объектов на изображении — это процесс обнаружения и классификации различных объектов или областей интереса на изображении. Существует несколько методов детекции объектов, каждый из которых имеет свои преимущества и недостатки в зависимости от конкретной задачи и условий. Для детекции объектов на видеопотоке наиболее подходят методы, обеспечивающие высокую скорость обработки и точность результатов. В зависимости от конкретных требований и условий видеопотока, можно использовать следующие методы:

1. Методы на основе свёрточных нейронных сетей (CNN): Свёрточные нейронные сети, такие как YOLO (You Only Look Once) и SSD (Single Shot MultiBox Detector), обладают высокой скоростью обработки и способны обнаруживать объекты в реальном времени на видеопотоке. Они являются хорошим выбором для задач, где требуется быстрая детекция объектов.
2. Методы на основе глубокого обучения с использованием регионов (R-CNN, Fast R-CNN, Faster R-CNN): Эти методы также предлагают высокую точность и скорость обработки, основанную на обработке регионов изображения. Они подходят для детекции объектов на видеопотоке с высокой степенью детализации.
3. Методы на основе каскадов Хаара и адаптивных каскадов: Методы, основанные на каскадах Хаара, такие как метод Viola-Jones, обеспечивают быструю детекцию объектов на видеопотоке с низкой вычислительной сложностью. Они могут быть полезны в случаях, когда требуется высокая скорость обработки при достаточной точности детекции.
4. Методы на основе дескрипторов объектов: Локальные дескрипторы объектов, такие как SIFT, SURF, ORB, могут использоваться для детекции объектов на видеопотоке, особенно в случаях, когда объекты имеют высокий уровень детализации или изменчивую текстуру.

Задача детекции достаточно изученная и популярная задача, от того существует множество различных методов. Наша задача состоит в детекции целого ряда объектов на видеопотоке в режиме реального времени. Наиболее предпочтительным для решения этой задачи являются свёрточные нейросети по нескольким причинам:

1. Высокая скорость обработки: Современные архитектуры свёрточных нейронных сетей, такие как YOLO (You Only Look Once) и SSD (Single Shot MultiBox Detector), оптимизированы для быстрой обработки видеопотоков. Они способны обнаруживать объекты в реальном времени с высокой скоростью, что особенно важно для приложений, требующих непрерывного мониторинга.
2. Высокая точность детекции: свёрточные нейросети демонстрируют высокую точность детекции объектов благодаря своей способности извлекать и анализировать признаки изображения на разных уровнях абстракции. Это позволяет им эффективно обнаруживать объекты различных размеров и форм в разнообразных условиях освещения и фона.
3. Способность к обучению на больших наборах данных: свёрточные нейросети позволяют обучать модели на больших объемах размеченных данных, что способствует их адаптации к разнообразным сценариям и условиям работы.
4. Возможность интеграции с графическими процессорами (GPU): Использование графических процессоров для параллельной обработки данных позволяет значительно ускорить вычисления, что делает их идеальным выбором для решения задач детекции на видеопотоке в реальном времени.
5. Гибкость и масштабируемость: Архитектуры свёрточные нейросети можно адаптировать под конкретные требования и условия задачи, а также масштабировать для работы с видеопотоками разного разрешения и качества.

Исходя из этих преимуществ, методы на основе свёрточных нейронных сетей являются оптимальным выбором для решения задачи детекции на видеопотоке в реальном времени, обеспечивая высокую скорость, точность и гибкость работы.

Выбор же именно модели свёрточной нейросети YOLO, а не SSD обусловлен более простотой архитектурой для первого знакомства с нейросетями, а также приятным и активным сообществом, сформированным вокруг архитектуры, что значительно облегчает обучение и решение самых разных проблем. Кроме того, YOLO хорошо справляется с детекцией объектов различных размеров на изображении, включая как крупные, так и мелкие объекты, благодаря механизму деления изображения на сетку и применению множества анкоров, что позволяет начинать строить картину окружающего мира задолго до необходимости взаимодействовать с объектами. Пример работы YOLO можно увидеть на рисунке 6



Рисунок 6 – Пример работы YOLO.

В 2024 году наиболее актуальной версией YOLO считается YOLOv8. Она обладает следующими преимуществом в виде лёгкой архитектуры, которая фокусируется на скорости и эффективности, а также имеет новые функции, такие как пользовательские привязки и трансферное обучение, которые делают модель проще в обучении и настройке для решения конкретных задач.

Также стоит сказать, что YOLOv9 и YOLOv6 также являются актуальными в наше время. Однако YOLOv9 хоть и превосходит YOLOv8 по точности, но имеет более медленное время вывода, а YOLOv6 лучше работает с небольшими объектами, но имеет более высокую вычислительную задержку, что не подходит для нашей конкретной задачи детекции.

## **4.3 Подготовка датасета и обучение модели**

Так как камера во время движения автопилота будет находится на носу трамвая, то и требуемый для обучения нейросети датасет тоже будет формироваться из записей с камеры в том же положении. Пример кадра с камеры можно увидеть на рисунке 7. Так как симулятор TramSim является обычной видеоигрой, то запись датасета происходила достаточно легко – с помощью наигранных часов и записи экрана. В дальнейшем из видеозаписи игрового процесса выбирались кадры, которые были достаточно богаты различными игровыми объектами.



Рисунок 7 – Вид камеры на носу.

В итоге из, суммарно, около часа записей игрового процесса было получено 109 различных кадров, готовых к разметке. Так как для решения задачи детекции использовалась модель YOLO, то для разметки было достаточно выделить на кадрах искомые объекты в прямоугольники, подписав каждый из них. Для удобства выполнения этого монотонного процесса было решено воспользоваться интернет-сервисом Roboflow.

Roboflow — это платформа, предназначенная для упрощения процесса подготовки данных для обучения моделей машинного обучения. Она предлагает различные инструменты и функции, которые помогают исследователям данных, разработчикам и инженерам в создании и управлении датасетами. Roboflow достаточно большой сервис, он обладает следующими возможностями:

1. Преобразование данных: позволяет быстро преобразовывать и аугментировать изображения и другие данные для обучения моделей, включая изменение размеров, повороты, изменение контрастности и многое другое.
2. Аннотирование данных: предоставляет средства для разметки изображений и других типов данных, необходимых для обучения моделей, таких как метки объектов на изображениях.
3. Интеграция с различными фреймворками: поддерживает интеграцию с популярными фреймворками машинного обучения, такими как TensorFlow, PyTorch, Keras и другими.
4. Управление датасетами: позволяет организовывать и управлять датасетами, включая загрузку, хранение и обмен данными.
5. Обучение моделей: предоставляет возможность обучения моделей машинного обучения на подготовленных данных.
6. Коллаборация: предоставляет средства для совместной работы над проектами и обмена данными и результатами между членами команды.

Roboflow помогает упростить и автоматизировать рутинные процессы, связанные с подготовкой данных для машинного обучения, что позволяет исследователям и разработчикам сосредоточиться на создании и улучшении моделей. Благодаря возможностям подключения дополнительных людей в команду разметки и обучения моделей прямо в их сервисе, не было страха того, что исследования могут уткнуться в потолок и занять очень много времени. К счастью, опасения не подтвердились и Roboflow был использован только как сервис для хранения датасета и последующего аннотирования.

Кроме того, прямо в сервисе имеющийся датасет можно было привести в форму, необходимую для обучения YOLO, а также применить аугментирование, расширив имеющийся датасет до 257 фотографий. Изображение датасета можно увидеть на восьмом рисунке. Там же датасет был разделён на тренировочный, валидирующий и тестирующий наборы в соотношении 86 к 10 к 4. Аугментирование позволило получить достаточно большой датасет для обучения, которого вполне хватило для обучения автопилота, а разделения данных на наборы улучшило результаты.

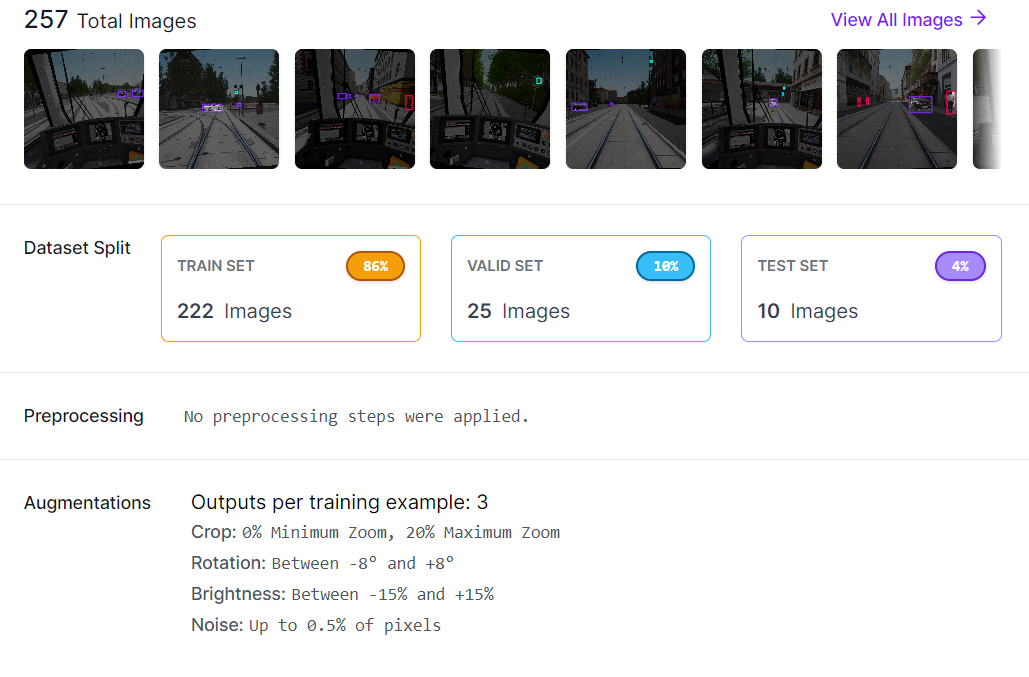


Рисунок 8 – Датасет на странице Roboflow.

В итоге, имеющийся датасет был использован для дообучения yolo8n, изначально обученной на стандартном датасете coco128. Версия nano была выбрана в виду того, что от модели требовалось моментально реагировать на происходящие события на дороге трамвая, точность определения объекта была на втором месте по важности.

Теперь требовалось обучить нейросеть, для этого использовался собственный ноутбук с видеокартой RTX 3050 Ti Laptop. Облачные сервисы не применялись в виду долгой настройки и вполне подходящего под задачу оборудования. В результате экспериментов по обучению сети, лучшей версией, согласно метрикам precision, recall и mAP50, была выбрана модель, обученная на 60 эпохах на имеющемся датасете. Именно это количество эпох оптимально для того, чтобы сеть максимально обучилась, но не имела проблем на валидирующем и тестировочном наборах. Результаты обучения представлены на рисунке 9.

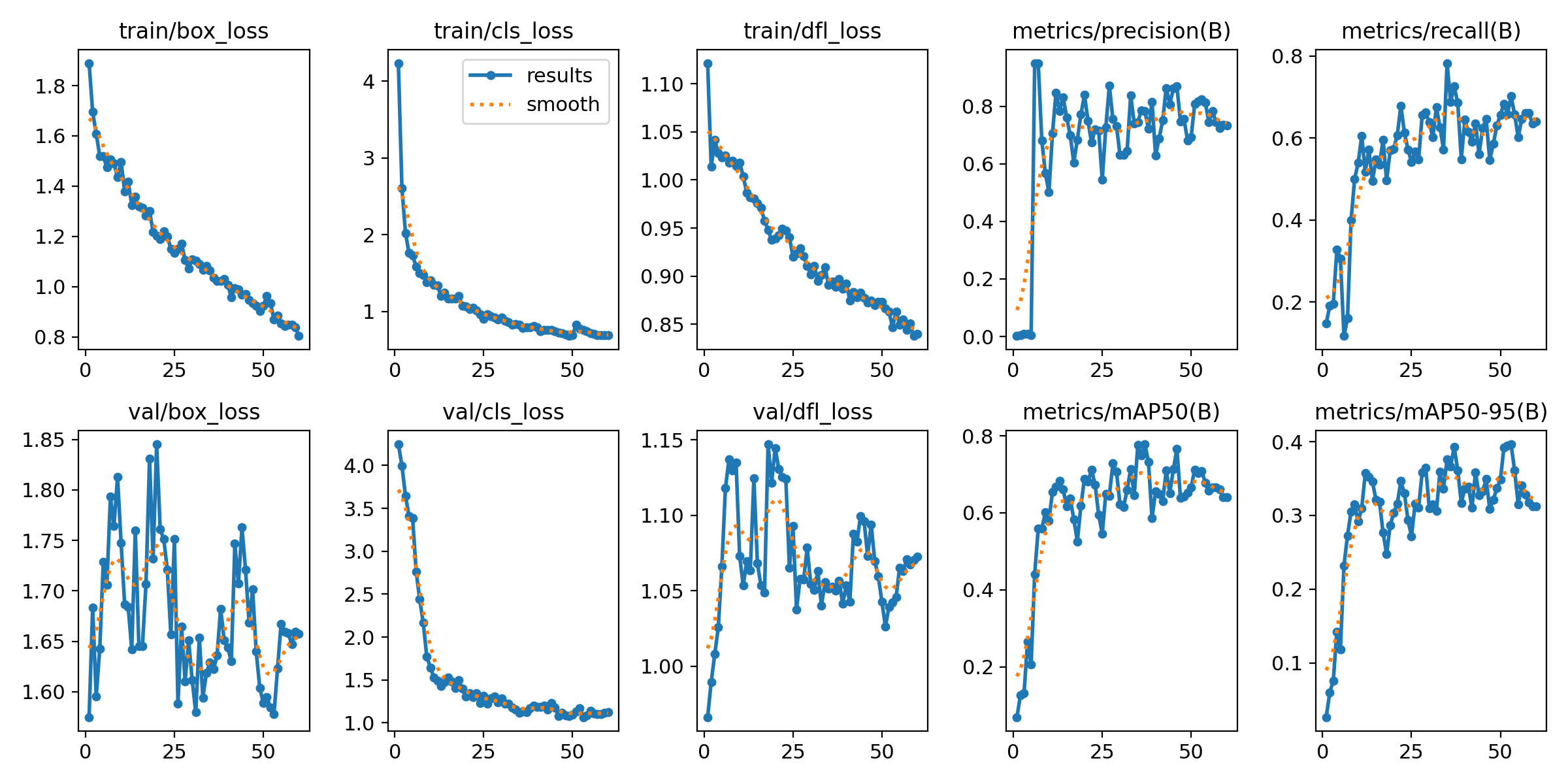


Рисунок 9 – Результаты обучения YOLOv8 на 60-ти эпохах.

# **5 Разработка модуля принятия решения**

## **5.1 Обработка размеченного кадра из YOLO**

В качестве интернет-среды как источника цифровых следов была выбрана онлайн-платформа для создания социальных отношения между людьми – социальная сеть. Однако существующие социальные сети хоть и предоставляют API («программный интерфейс приложения») для разработчиков, но из соображений безопасности не позволяют получить доступ к активным цифровым следам другого пользователя. Из-за этого процессы разработки системы прогнозирования, и её тестирования оказываются чересчур трудоемкими занятиями. Так же затруднителен вариант со сбором информации напрямую с чужих страниц из-за следующих двух ограничений.

1. Такая информация ограничивается мерами конфиденциальности, заложенными при создании той или иной социальной сети. К примеру, ни в одной социальной сети нельзя без участия пользователя просмотреть список его диалогов;
2. В большинстве социальных сетей присутствует скрытие содержимого страницы от посторонних глаз.

Исходя из перечисленных ограничений, было принято решение разработки собственной социальной сети с необходимым функционалом для возможности оставления цифровых следов.

## **5.2 Программная реализация серверной части социальной сети**

Серверная часть отвечает за связь компонентов приложения друг с другом: организует передачу данных с клиента в базу данных и наоборот, организует передачу активных и пассивных следов клиента с базы данных в модули, ответственные за обработку, организует передачу экспертной системе совокупности ответов модулей, ответственных за обработку активных и пассивных следов и организует передачу от экспертной системы клиенту ответа системы. Общая структура приложения представлена на рисунке 6.

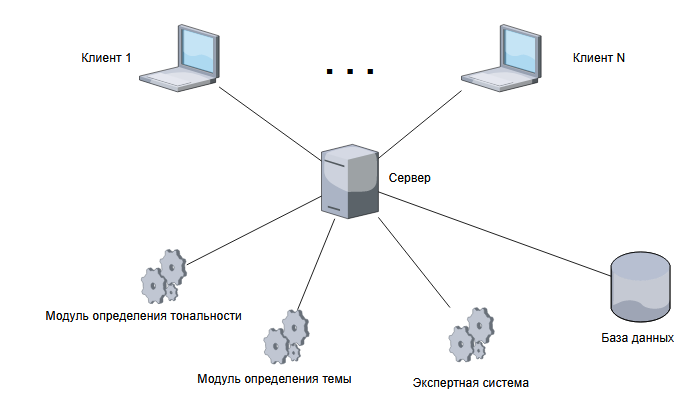


Рисунок 6 – Общая структура приложения.

В качестве платформы для серверной части приложения был выбран NodeJS, в качестве системы управления базами данных приложения была выбрана СУБД Oracle. В качестве протокола для общения между пользователями, такими как отправка сообщений, выставление лайков, подписки и т.д., используется протокол WebSocket.

## **5.3 Программная реализация клиентской части социальной сети**

Клиентская часть приложения реализована с помощью веб-технологий HTML, CSS, JS, а также библиотеки DevExtreme.

В качестве примера интерфейса рассмотрим несколько окон. Окно авторизации, представленное на рисунке 7, предоставляет интерфейс авторизации в систему. В случае, если не существует действующего аккаунта, пользователю предлагается пройти регистрацию.

Изображение выглядит как текст, визитка, снимок экрана

Автоматически созданное описание

Рисунок 7 – Окно авторизации.

Окно диалогов, представленное на рисунке 8, содержит список всех существующих диалогов пользователя, поиск нужного диалога, а также создание нового диалога и удаление существующего. При выборе диалога из списка всех диалогов открывается панель, содержащая историю сообщений между пользователем и вторым участником диалога и блок написания нового сообщения.

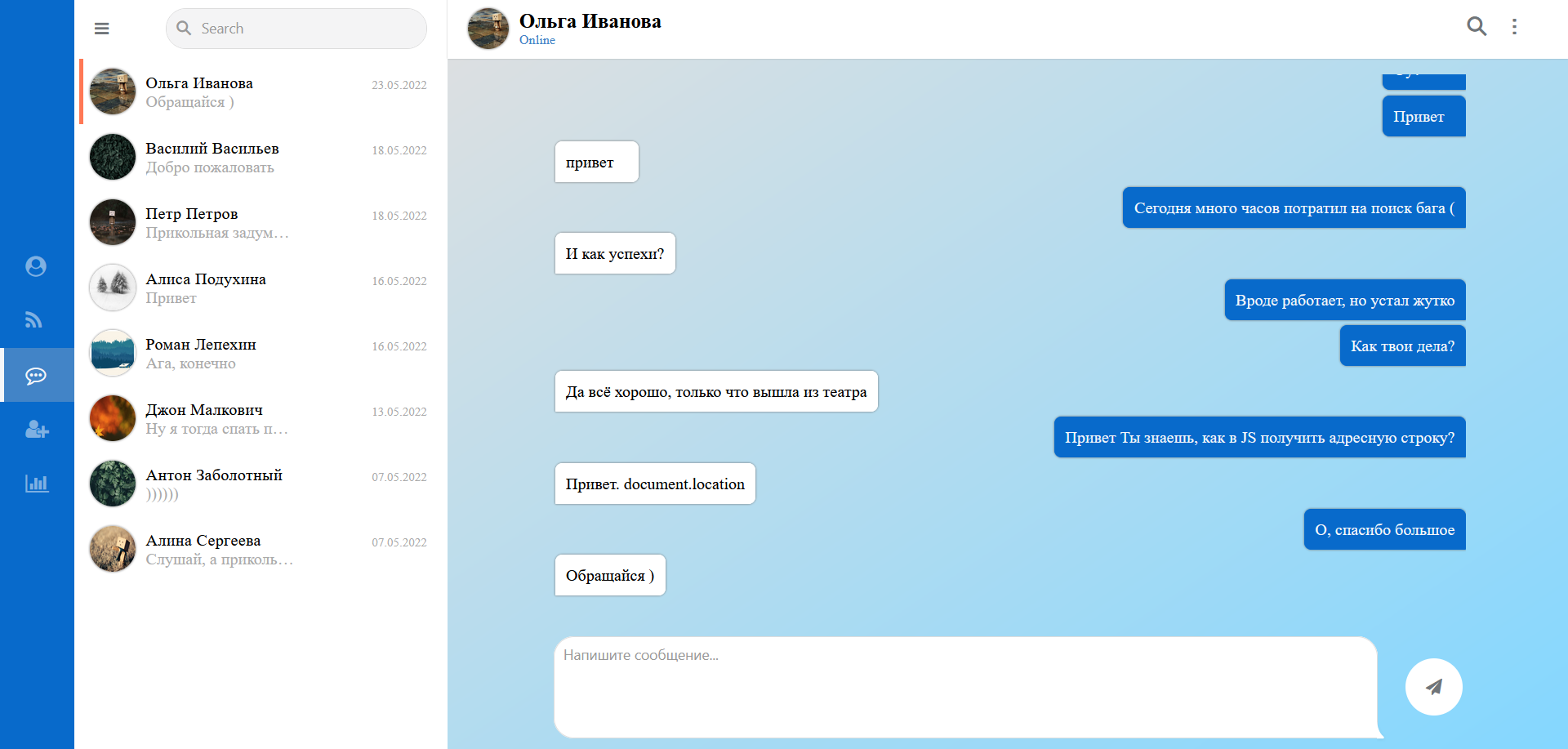


Рисунок 8 – Окно диалогов.

Окно вывода прогноза профессии и статистики пользователя, представленное на рисунке 9, содержит статистику активных и пассивных следов пользователя, представленную в виде диаграмм доли сообщений по темам и доли лайков по темам, а также статистику по странице: количество друзей, количество фотографий и так далее. Содержит ответ экспертной системы на основании активных и пассивных следов пользователя.

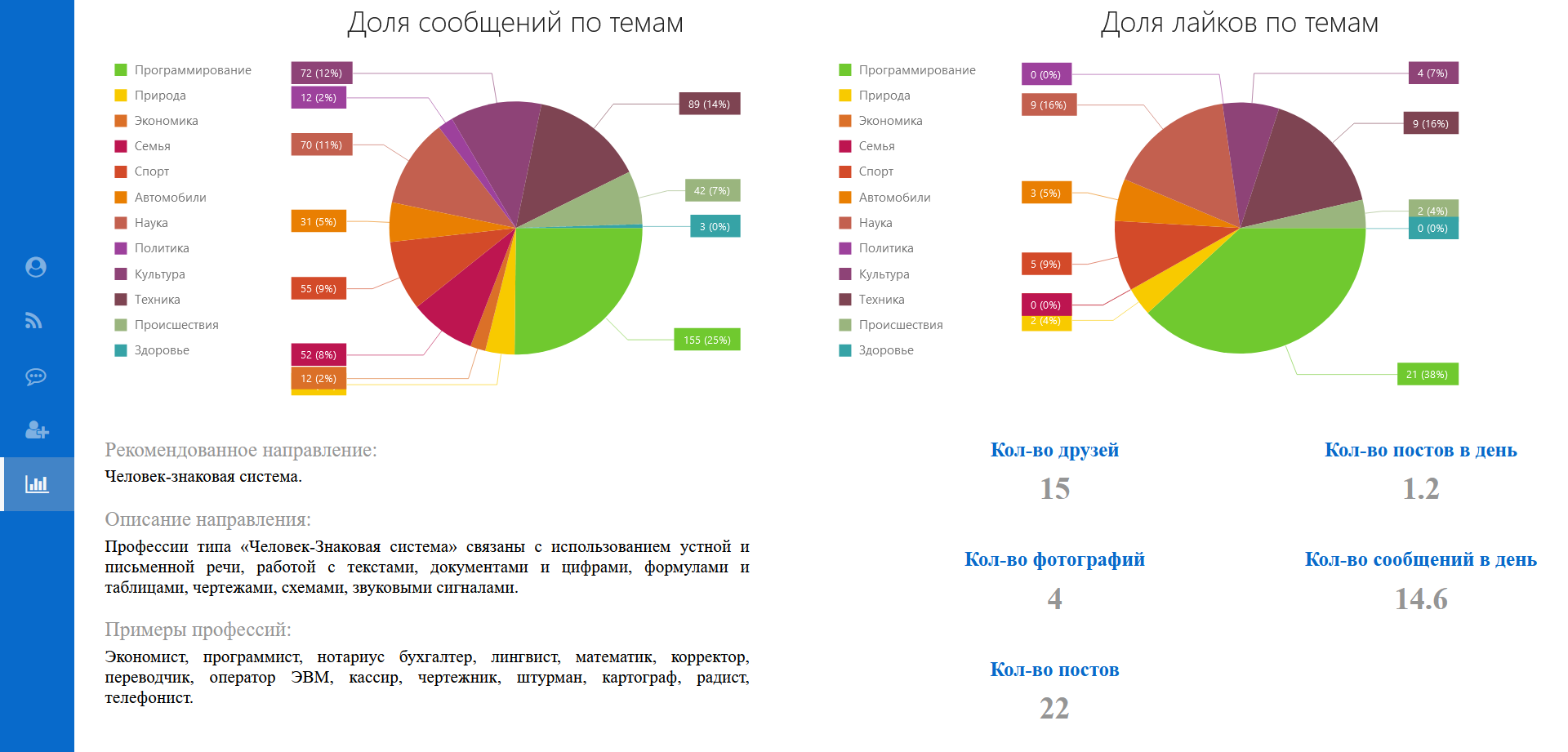


Рисунок 9 – Окно вывода прогноза по профориентации.

# **6 Разработка системы Data Mining**

## **6.1 Инструменты анализа данных**

Одним из наиболее популярных языков программирования для анализа данных является Python. Для него создано большое количество библиотек и фреймворков. В работе используется среда разработки PyCharm. Для визуализации в работе используются библиотеки matplotlib и sklearn, хранение данных осуществляется в оперативной памяти с помощью библиотеки pandas.

## **6.2 Набор данных для обучения моделей**

### **6.2.1 Набор данных для обучения моделей классификации тональности текста**

В качестве начальных и проверочных данных используется набор SST. Он состоит из 11855 полученных из обзоров фильмов предложений с метками настроения от 1 до 5:

* 1 – крайне отрицательный;
* 2 – слабо отрицательный;
* 3 – нейтральный;
* 4 – слабо положительный;
* 5 – крайне положительный.

А также из 215 000 фраз, которые составляют предложения в наборе. Пример данных изображен на рисунке 10.

Изображение выглядит как текст, компьютер, монитор, рабочий стол

Автоматически созданное описание

Рисунок 10 – Фрагмент набора данных для классификации

### **6.2.2 Набор данных для обучения моделей классификации тем документов**

В качестве набора данных для обучения моделей классификации тем документов были собраны тексты с новостных порталов, имеющие определенную категорию. Так же в ходе анализа различных ресурсов были вычленены абзацы, которые наибольшим образом относятся к той или иной предметной области. После процесса сбора текстов был проведен процесс лемматизации и удаления стоп-слов. Всего набор содержит 3562 размеченных текста, где метка соответствует одному из следующих классов:

* Здоровье;
* Программирование;
* Спорт;
* Автомобили;
* Экономика;
* Культура;
* Политика;
* Техника;
* Происшествия;
* Семья;
* Природа;
* Наука.

На рисунке 11 представлено распределение текстов по темам.



Рисунок 11 – Количество текстов по каждой теме.

Фрагмент набор данных представлен на рисунке 12.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 12 – Фрагмент набора данных.

## **6.3 Разработка системы классификации тональности текста**

На вход системе подается массив сообщений пользователя. Ответом системы является размеченный массив сообщений, где меткой выступает тональности сообщения.

Тестирование и обучение классификаторов проводилось на одних и тех же данных.

Для визуализации результатов классификатора использовалась матрица ошибок. Матрица ошибок сводит в таблицу процент верных предсказаний для каждого класса, из-за чего становится легче увидеть, какие классы наименее точно предсказаны данным классификатором. В идеале классификатор должен получить почти 100% правильных предсказаний, следовательно все элементы вне побочной диагонали должны быть стремиться к нулю, как на рисунке 13.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 13 – Пример матрицы ошибок.

Начнем анализ с самого простого классификатора из рассматриваемых: классификатор на основе метода TextBlob. Классификатор показывает следующие результаты предсказания, изображенные с помощью матрицы ошибок на рисунке 14.

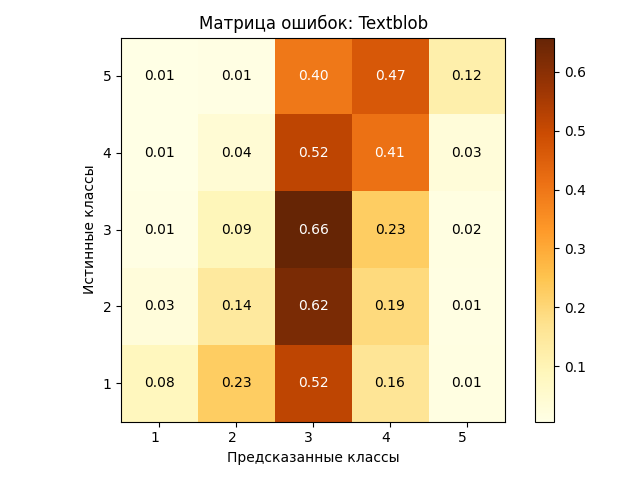


Рисунок 14 – Матрица ошибок предсказания TextBlob.

На рисунке 15 покажем точность и F-меру полученных предсказаний.

Изображение выглядит как текст, снимок экрана, компьютер, внутренний

Автоматически созданное описание

Рисунок 15 – Оценка TextBlob.

Метод, работающий на правилах и словарях, показал плохую точность, а также F-меру, что говорит о том, что классификатор не мог точно определять, к какому из 5 классов принадлежит рассматриваемое предложение. Из матрицы ошибок видно, что классификатор с трудом отделяет нейтральный класс от всех остальных. Это связано с тем, что классификатор на основе TextBlob пытается дать результат, основанный на прямом результате анализа грамматики: он не пытается проверить, является ли все предложение отрицательным или нет, а определяет тональность отдельных слов и после усредняет значение для определения тональности всего предложения. И в результате усреднения большинство предложений получили нейтральную окраску.

Рассмотрим классификатор, построенный на основе VADER, который так же работает на правилах. Классификатор показывает предсказания, изображенные на рисунке 16.

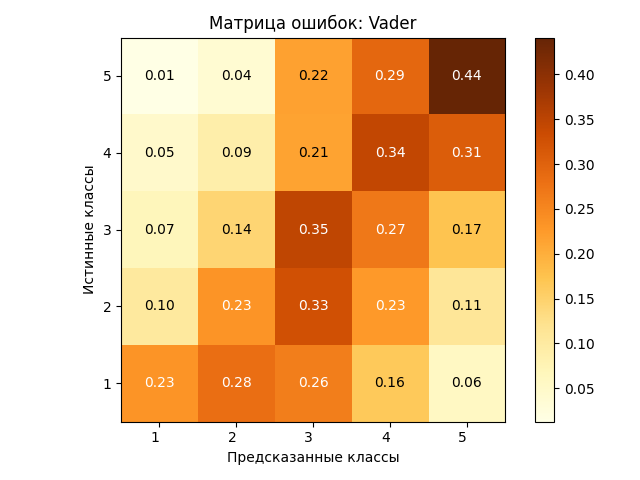


Рисунок 16 – Матрица ошибок предсказания VADER.

На рисунке 17 покажем точность и F-меру классификатора.

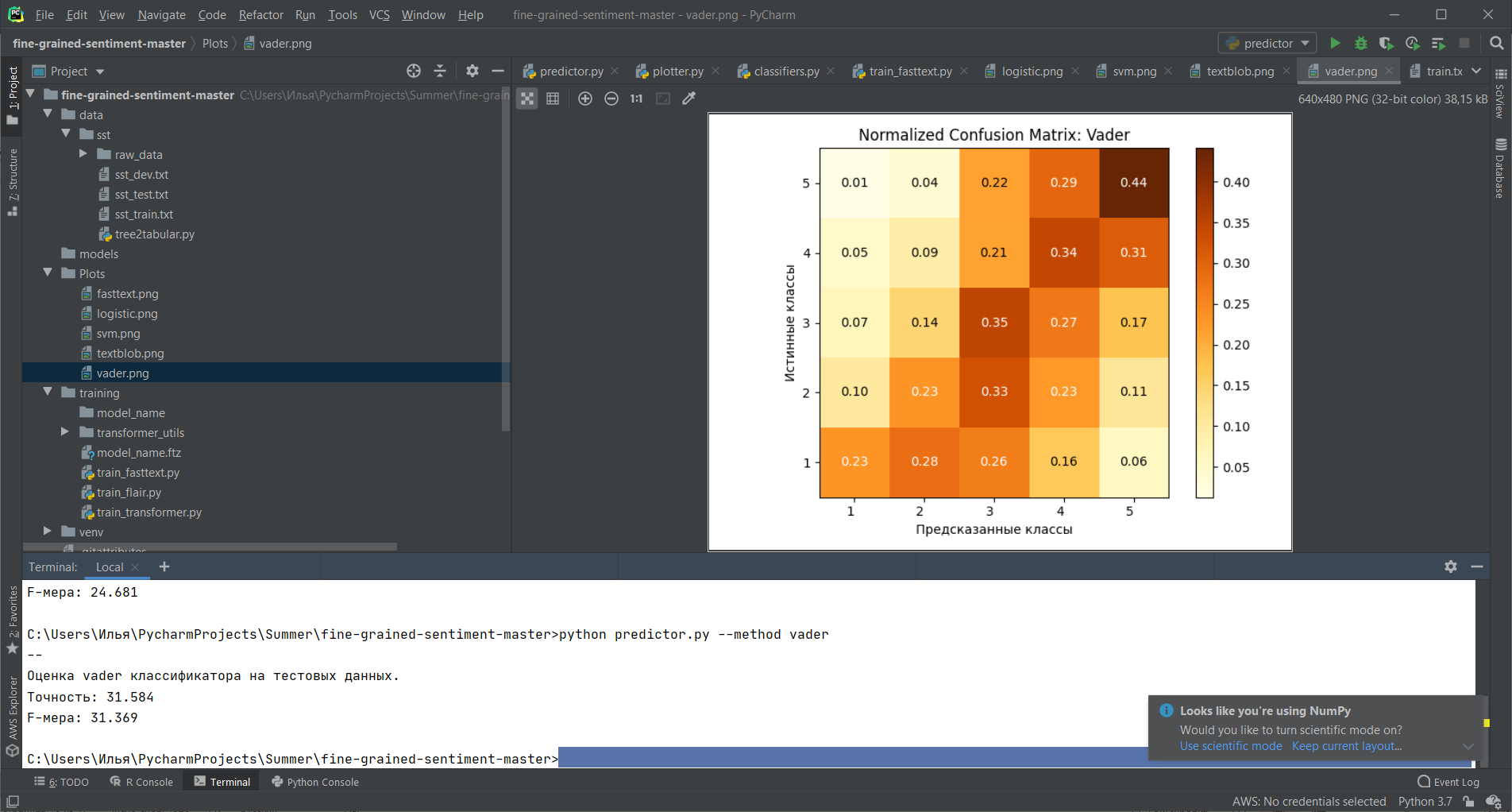


Рисунок 17 – Оценка VADER.

Классификатор на основе VADER так же, как и предыдущий классификатор, усредняет тональности всех слов в предложении, но VADER разрабатывался с акцентом на тексты социальных сетей, а значит в нем много внимания уделяется правилам, которые отражают суть текста, обычно встречаемого в социальных сетях: короткие предложения со смайликами, повторяющиеся слова и обильное использование знаков препинания. Этим и объясняется большой разброс вне побочной диагонали: очень низкие или очень высокие составные оценки присваиваются тексту, который имеет много капитализации, пунктуации, повторения слов и смайликов. Тем не менее классификатор на основе VADER показывает лучшие результаты, чем классификатор на основе TextBlob, что можно увидеть на рисунке 18.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18 – Сравнение предсказаний классификаторов.

Недостатком методов, основанных на словарях и правилах, является большая затратность на составление этих самых словарей и правил, а также неспособность правильно обрабатывать незнакомые конструкции и слова, а в условиях постоянно развивающегося языка это большой минус.

От подходов, основанных на правилах и словарях, перейдем к машинному обучению с учителем. Рассмотрим классификатор на основе логистической регрессии. Классификатор показывает предсказания, изображенные на рисунке 19.

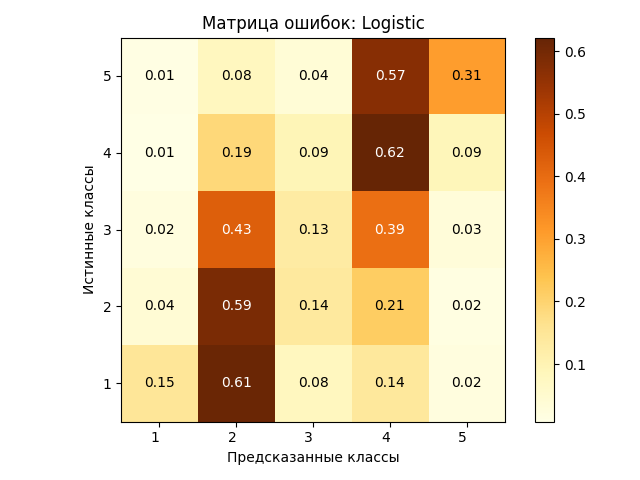


Рисунок 19 – Матрица ошибок логистической регрессии.

На рисунке 20 покажем точность и F-меру классификатора.

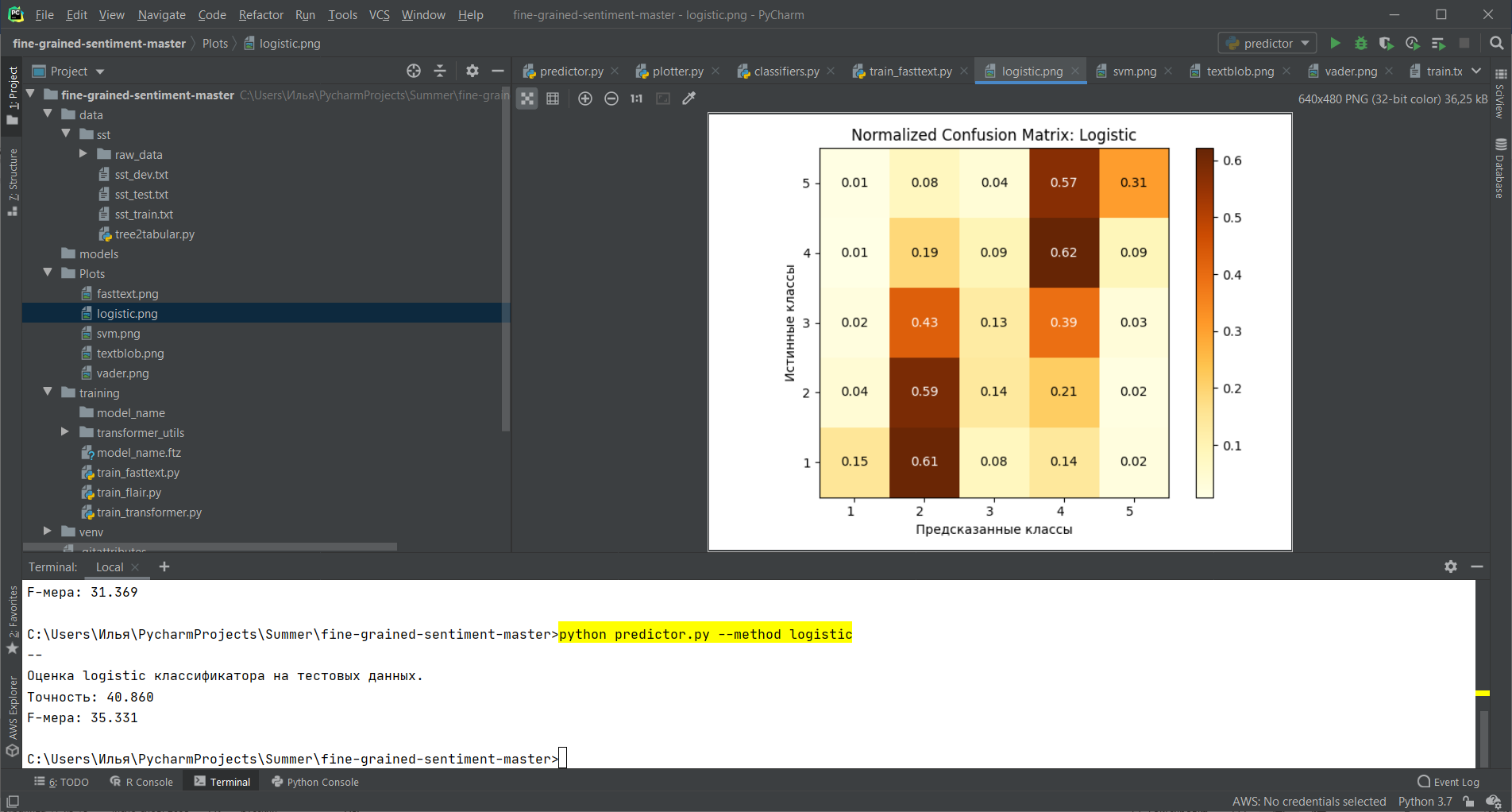


Рисунок 20 – Оценка логистической регрессии.

Сравним результаты классификатора на основе логистической регрессии с результатами предыдущих классификаторов на рисунке 21.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 21 – Сравнение предсказаний классификаторов.

Как видим, машинное обучение с учителем показывает себя гораздо лучше методов на словарях и правилах, однако в случае с логистическая регрессией результаты буду значительно лучше, если данные будут линейно разделяемы, а в случае с 5 классами добиться этого крайне сложно. Протестируем классификаторы на двухзначной шкале тональности, а именно отнесем предложения к отрицательному и положительному классам. На рисунке 22 покажем матрицу ошибок для классификатора на основе TextBlob.

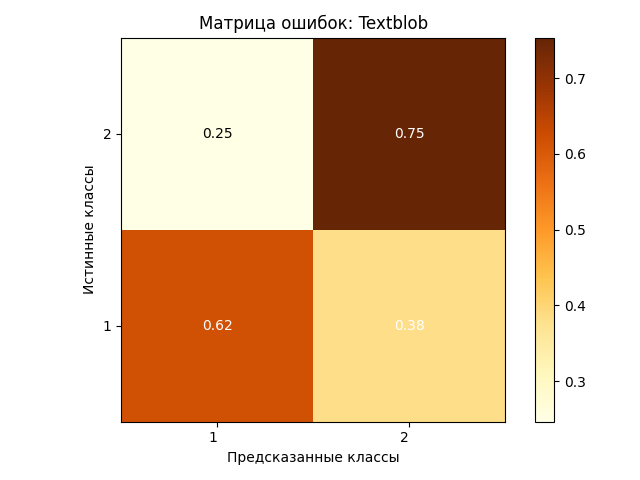


Рисунок 22 – Матрица ошибок TextBlob.

На рисунке 23 покажем матрицу ошибок для классификатора на основе VADER.

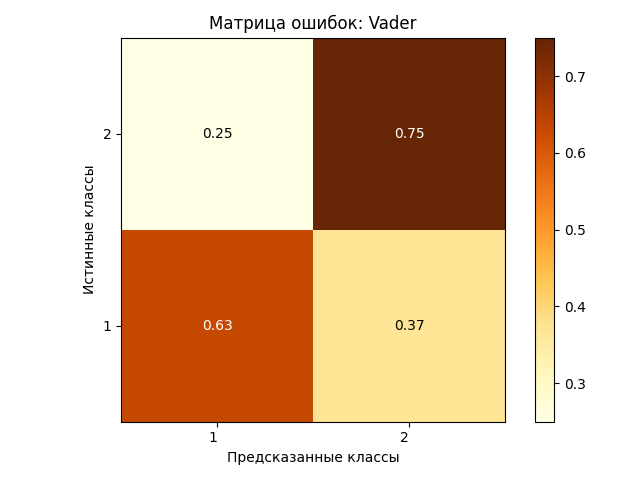


Рисунок 23 – Матрица ошибок VADER.

На рисунке 24 покажем матрицу ошибок для классификатора на основе логистическая регрессии.

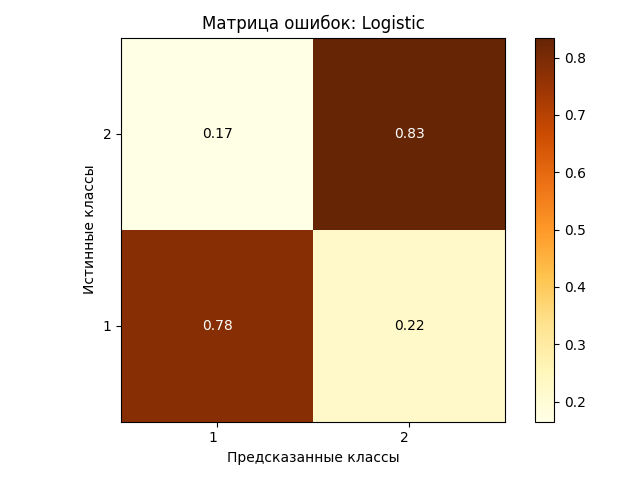


Рисунок 24 – Матрица ошибок логистическая регрессии.

И сравним результаты классификаторов на рисунке 25.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 25 – Сравнение предсказаний классификаторов.

Логистическая регрессия смогла отнести к правильным классам 80 процентов входных предложений, что является хорошим результатом в задаче определения тональности текста и по этому показателю она превосходит классификаторы на основе TextBlob и VADER.

Если же мы посмотрим на другой рассматриваемый метод машинного обучения с учителем – метод опорных векторов, то на рисунке 26 увидим превосходство логистической регрессии в точности в задаче бинарной классификации.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 26 – Сравнение предсказаний классификаторов.

Но также на рисунке 27 увидим улучшение этого показателя у метода опорных векторов перед логистической регрессией в задаче отнесения к 5 классам. Действительно SVM поддерживает как линейные, так и нелинейные решения, к тому же он лучше справляется с выбросами.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 27 – Сравнение предсказаний классификаторов.

На рисунке 28 отобразим матрицу ошибок для SVM.



Рисунок 28 – Матрица ошибок SVM.

Процент правильно определенных нейтральных предложений все ещё низок, но в сравнении с предыдущими классификаторами SVM точнее относит предложения к крайне отрицательным и крайне положительным классам.

У классификатора, основанного на методе FastText, есть ряд преимуществ в сравнении с предыдущими рассматриваемыми. Иерархический метод классификации, обладающий большой информационной ёмкостью. В качестве модели векторных представлений применяется skip-gram с негативным сэмплированием. Но главным преимуществом выступает добавление к основной модели subword-модели. В такой модели слова представляются через цепочки символов.

Такой подход позволяет модели работать с ранее не встречаемыми словами. Совокупность этих преимуществ выливается в самую большую точность для задачи отнесения предложения к одному из пяти классов из рассматриваемых классификаторов, изображенную на рисунке 29.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 29 – Сравнение предсказаний классификаторов.

На матрице ошибок, изображенной на рисунке 30, видно, что классификатор гораздо точнее определяет предложения, имеющие нейтральную окраску, что и дает более высокую точность.

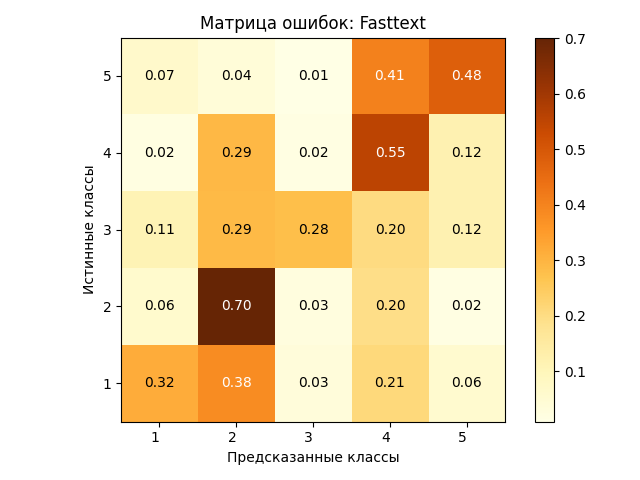


Рисунок 30 – Матрица ошибок FastText.

Для удобства восприятия покажем сравнения классификатор в виде столбчатых диаграмм для задачи классификации с 5 классами на рисунке 31 и для задачи классификации с 2 классами на рисунке 32.



Рисунок 31 – Сравнение классификаторов (2 класса).

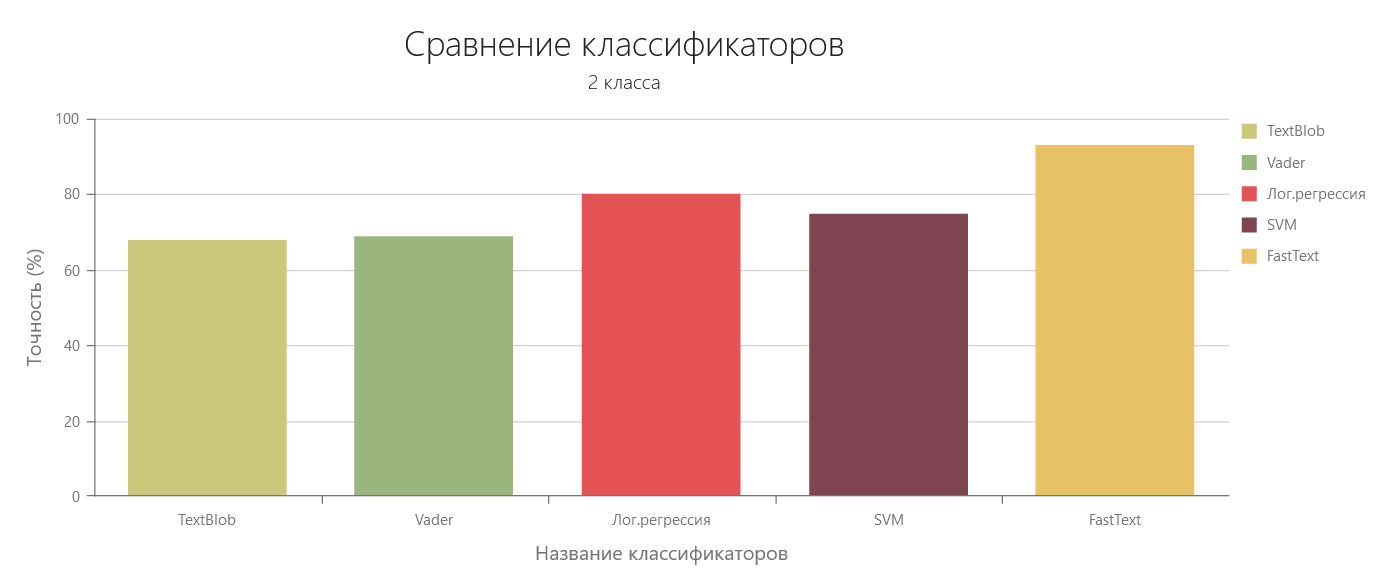


Рисунок 32 – Сравнение классификаторов (5 классов).

## **6.4 Разработка системы классификации тем документов**

На вход системе подается массив сообщений пользователя. Ответом системы является размеченный массив сообщений, где меткой выступает один из 12 класс, описанных в пункте 6.2.2.

Для выбора классификатора необходимо провести сравнительный анализ точности предсказаний.

Первым делом рассмотрим алгоритм Random Forest – алгоритм машинного обучения, заключающийся в использовании ансамбля решающих деревьев, каждое из которых само по себе даёт низкое качество классификации, но за счёт их количества качество классификации повышается. Пример работы алгоритма представлена на рисунке 33.

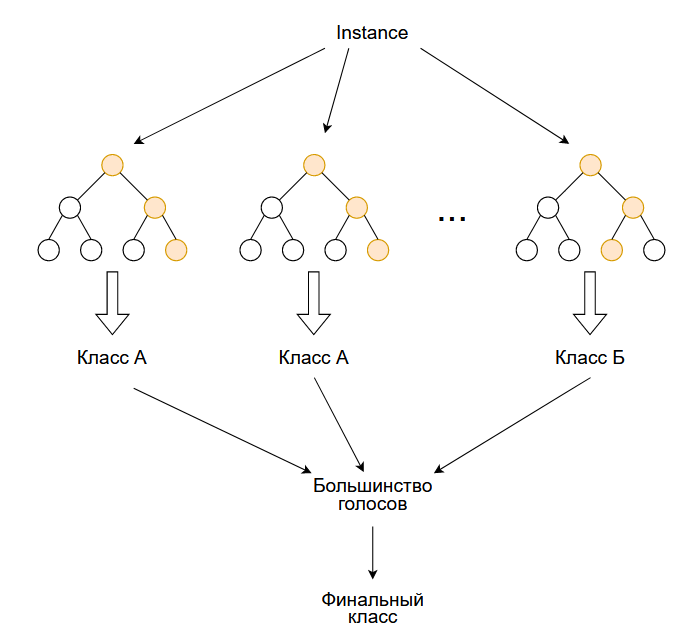


Рисунок 33 – Пример работы алгоритма Random Forest.

Следом рассмотрим алгоритм K-nearest neighbors (KNN). В случае использования алгоритма для классификации объекту присваивается тому классу, который является наиболее распространённым среди k соседей данного элемента, классы которых уже известны. Пример работы алгоритма представлен на рисунке 34.



Рисунок 34 – Пример работы алгоритма KNN.

Рассмотри алгоритм Support Vector Machine (SVM) – линейный алгоритм, используемый в задачах классификации. Основная идея метода – перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с наибольшим зазором в этом пространстве. Две параллельных гиперплоскости строятся по обеим сторонам гиперплоскости, разделяющей классы. Пример работы алгоритма представлен на рисунке 35.

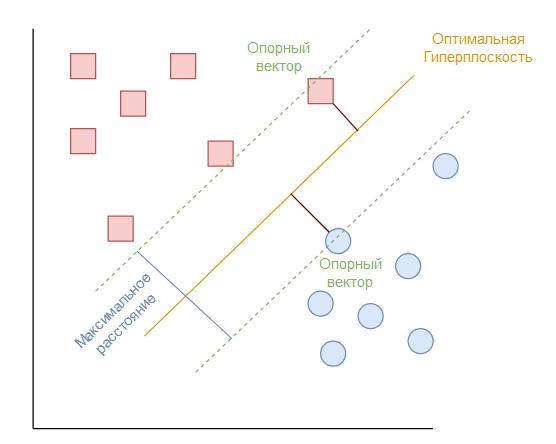


Рисунок 35 – Пример работы алгоритма SVM.

SGD – итерационный метод для оптимизации целевой функции. Будет использоваться для обучения метода опорных векторов.

Проведем сравнение точности описанных выше алгоритмов на тестовых данных. Модели обучались на одинаковых данных, в количестве 2800 текстов. Сравнение точности работы алгоритмов представлено рисунке 36.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 36 – Точность рассматриваемых алгоритмов.

Как видно из рисунка 36, лучшую точность, а именно 93 процента, в задаче классификации с 12-ю классами показал классификатор SGD, следовательно в дальнейшем в качестве классификатора документов по темам будет использоваться именно этот алгоритм. Пример работы классификатора SGD представлен на рисунке 37.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 37 – Пример работы классификатора SGD.

# **7 Разработка экспертной системы**

## **7.1 Построение экспертной системы**

Экспертная система на основе метода опроса включает в себя следующие шаги:

1. Выделение термов и области определения из нескольких объектов;
2. Построение для каждого терма функция принадлежности на основе косвенного метода попарных сравнений;
3. Проведение расчета весовых коэффициентов экспертов;
4. Реализация для каждого эксперта проверки согласованности его результатов на основании критерия непротиворечивости;
5. Реализация проверки согласованности мнений всех экспертов;
6. В случае несогласованности мнений исключить экспертов с самыми низкими весовыми коэффициентами до согласованности мнений.

В качестве области определения была выбрана классификация видов профессиональной деятельности Е.А. Климова, описанная в разделе 4, а именно:

* «Человек-Техника»;
* «Человек-Природа»;
* «Человек-Знаковая система»;
* «Человек-Художественный образ»;
* «Человек-Человек».

В качестве термов выступают вопросы в сравнительной форме, относящиеся к определенной предметной области, к социальной активности и поведению человека:

* Кто больше интересуется программированием;
* Кто больше интересуется природой;
* Кто больше интересуется политикой;
* Кто чаще выставляет фотографии;
* Кто больше общается;
* И т.д.

Для выставления оценок экспертам был проведен сравнительный анализ компетентности экспертов с помощью матрицы попарных сравнений.

## **7.2 Сбор оценок экспертов**

Для получения оценок каждому эксперту для заполнения были разосланы матрицы попарных сравнений для каждого выделенного терма, с прикрепленными правилами заполнения матрицы. В качестве примера рассмотрим заполненную одним из экспертов матрицу попарных сравнений для терма «Кто больше интересуется природой», представленную на рисунке 38.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 38 – Матрицы попарных сравнений для терма «Кто больше интересуется природой».

## **7.3 Реализация и тестирование экспертной системы**

После того, как были собраны все матрицы попарных сравнений и проведено оценивание компетентности экспертов, можно получить значения функций принадлежности для каждого терма [10].

Для начала необходимо рассчитать среднюю геометрическую в каждой строке матрицы попарных сравнений. Расчет производится по формуле (11).

После чего необходимо нормировать полученные значения по максимуму.

На рисунке 39 показан процесс выполнения шага.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 39 – Процесс нахождения средних геометрических.

После необходимо убедиться в согласованности экспертов. Процесс проверки согласованности состоит из следующих шагов.

Необходимо рассчитать компоненты нормализованного вектора приоритетов (НВП) по формуле (12).

Далее необходимо совершить проверку согласованности локальных приоритетов путем расчета трёх характеристик:

1. Собственного значения матрицы, описанного формулой (13);
2. Индекса согласования, описанного формулой (14);
3. Расчет обобщенного индекса согласования, описанного формулой (15);
4. Расчет обобщенного отношения согласованности, описанного формулой (16).

где ОПСС – обобщенный показатель случайной согласованности, зависящий от размера матрицы попарных сравнений, согласно таблице на рисунке 40.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 40 – таблица ПСС.

Оценки в матрице считаются согласованными, если , в противном случае нужно корректировать матрицы сравнения вариантов по критериям.

Покажем выполнения шага проверки согласованности.

На рисунке 41 изображены обобщенные отношения согласованности по каждому вопросу.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 41 – Пример несогласованности мнений экспертов.

Как видно из рисунка в третьем вопросе нет согласованности экспертов. Поэтому находим самого выделяющегося эксперта и удаляем его из списка экспертов по данному вопросу. Таким образом удается добиться согласованности, что видно на рисунке 42.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 42 – Пример получения согласованности мнений экспертов.

Теперь, когда мнения экспертов согласованны, необходимо умножить значения функции принадлежности на вектор оценок компетентности экспертов. Таким образом находятся значения функции принадлежности для рассматриваемой лингвистической переменной. Результат можно увидеть на рисунке 43, где номер элемента вектора соответствует одному из пяти рассматриваемых классов профессий.



Рисунок 43 – Пример значения функции принадлежности.

# **8 Схема работы системы прогнозирования**

Отобразим схему работы системы прогнозирования.



Рисунок 44 – Схема работы системы прогнозирования.

Как видно из рисунка 44 работа системы состоит в следующем:

* пользователь оставляет свои пассивные и активные следы в социальной сети;
* следы пользователя передаются на сервер, где они помещаются в базу данных;
* данные из базы данных передаются в системы Data Mining, в данной работе – классификация сообщений по темам, классификация контента в понравившихся записях по темам, классификация тональности текста;
* Обработанные данные передаются экспертной системе на основе нечеткого анализа, где формируются результаты;
* Происходит суммирование результатов системы с учетом значимости следов;
* На основании полученных результатов пользователю выдается прогноз профориентации.

# **9 Тестирование системы прогнозирования профориентации**

Для тестирования работы системы прогнозирования профориентации смоделируем поведение людей, которые предрасположены к разным направлениям.

Рассмотрим первого пользователя. На рисунке 45 представлено количество сообщений, отправленных пользователем.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 45 – Количество сообщений первого рассматриваемого пользователя.

На рисунке 46 можно увидеть, что большинство сообщений было написано на тему «программирование».

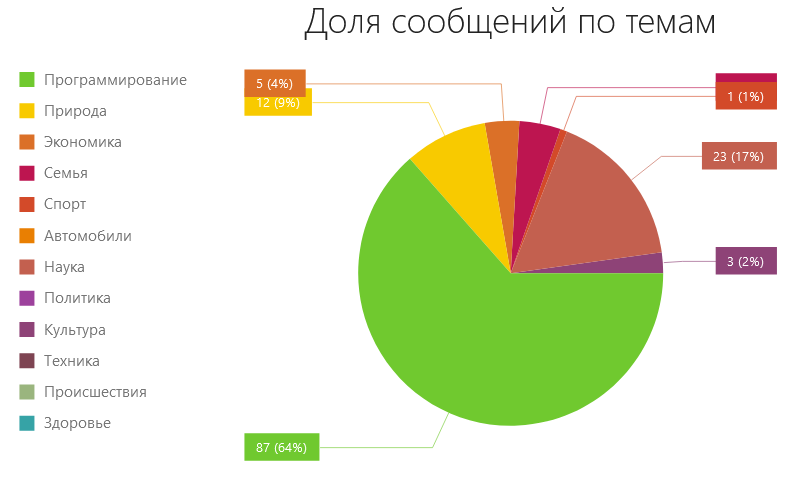


Рисунок 46 – Доля сообщений по темам первого рассматриваемого пользователя.

На рисунке 47 представлен фрагмент постов, понравившихся первому рассматриваемому пользователю.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 47 – Фрагмент постов, понравившихся первому рассматриваемому пользователю.

Всего пользователь оставил 42 лайка, что можно увидеть на рисунке 48.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 48 – Количество лайков, оставленных первым пользователем.

Большая часть постов содержит контент, относящийся к классу «программирование», что можно заметить на рисунке 49.

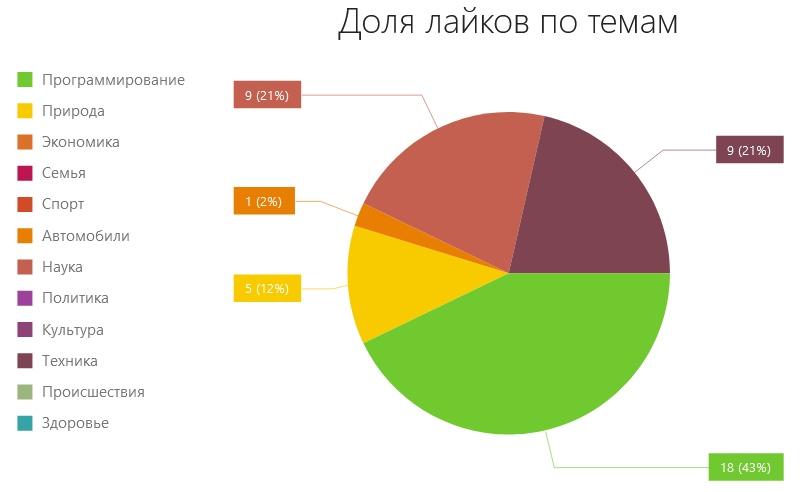


Рисунок 49 – Доля лайков, оставленных первым рассматриваемым пользователем, по каждой теме.

Исходя из активных следов, оставленных рассматриваемым пользователем, система выдала ответ, представленный на рисунках 50 и 51.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 50 – Ответ экспертной системы.

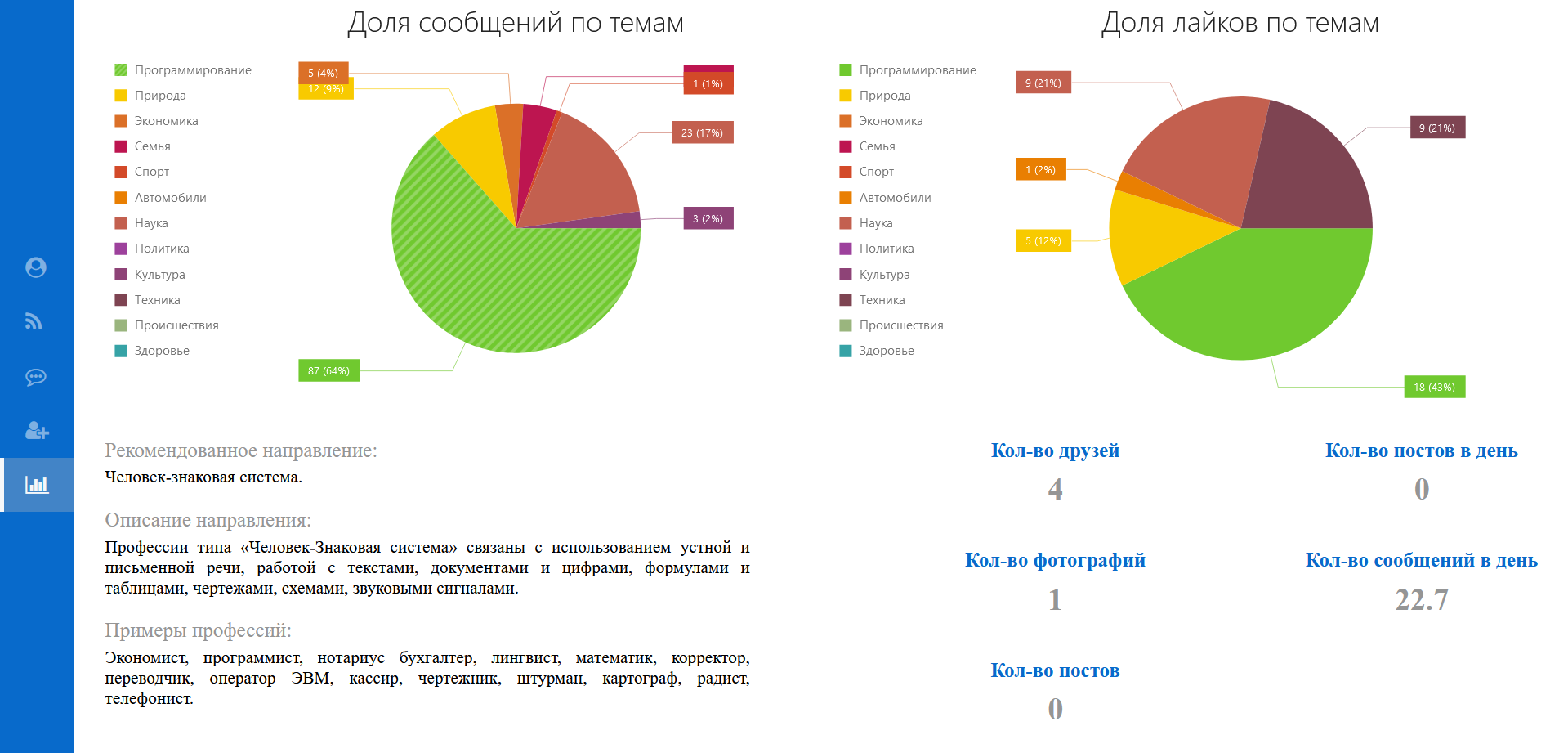


Рисунок 51 – Интерфейс ответа экспертной системы.

Рассмотрим второго пользователя. На рисунке 52 представлено количество сообщений, отправленных пользователем.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 52 – Количество сообщений второго рассматриваемого пользователя.

На рисунке 53 можно увидеть, что большинство сообщений было написано на тему «Природа».

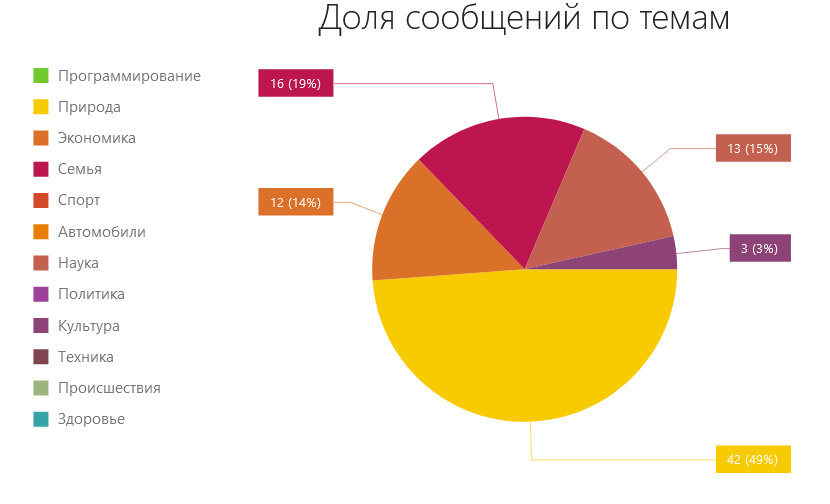


Рисунок 53 – Доля сообщений по темам второго рассматриваемого пользователя.

На рисунке 54 представлен фрагмент постов, понравившихся второму рассматриваемому пользователю.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 54 – Фрагмент постов, понравившихся первому рассматриваемому пользователю.

Всего пользователь оставил 31 лайк, что можно увидеть на рисунке 55.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 55 – Количество лайков, оставленных вторым пользователем.

Большая часть постов содержит контент, относящийся к классу «Наука», что можно заметить на рисунке 56.

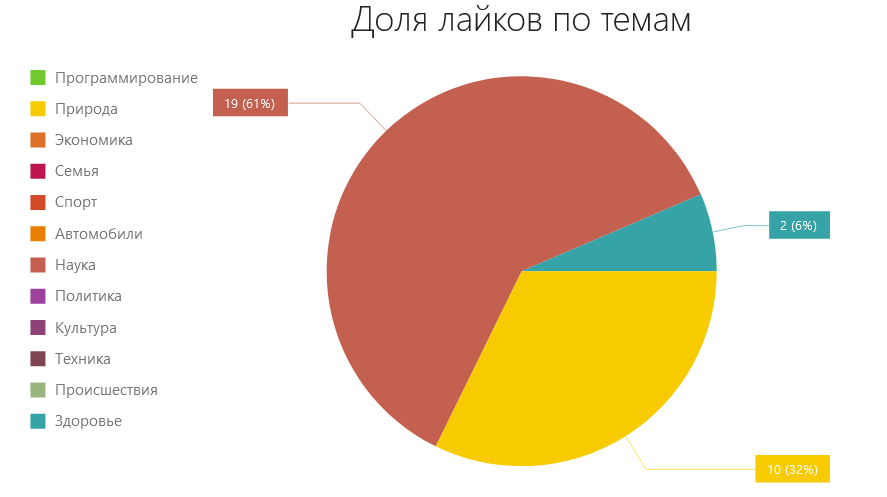


Рисунок 56 – Доля лайков, оставленных вторым рассматриваемым пользователем, по каждой теме.

Исходя из активных следов, оставленных рассматриваемым пользователем, система выдала ответ, представленный на рисунках 57 и 58.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 57 – Ответ экспертной системы.

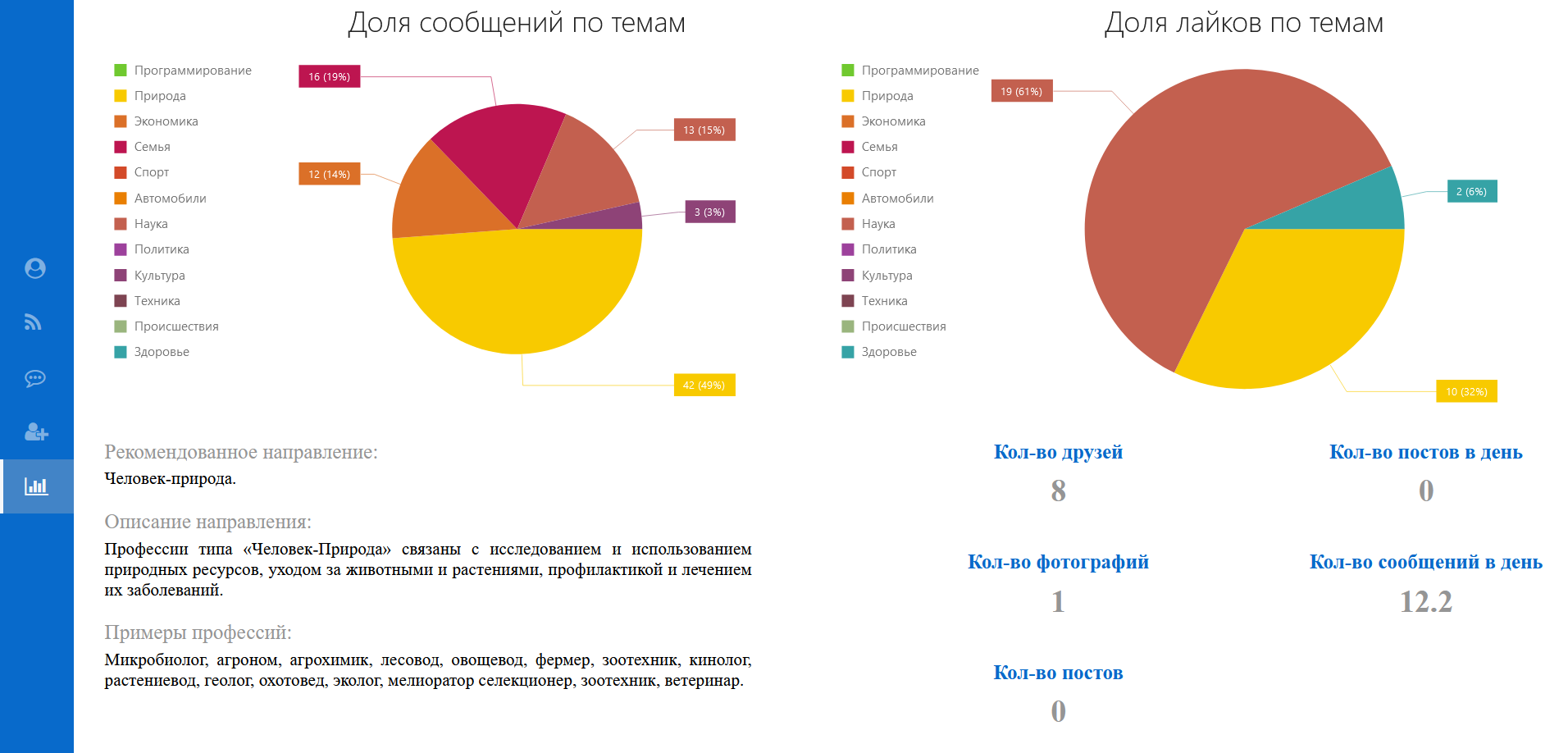


Рисунок 58 – Интерфейс ответа экспертной системы.

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения данной работы была реализована система прогнозирования профориентации человека по его цифровым следам в социальной сети на основе нечеткого анализа.

Данная система в разной степени учитывает такие активные и пассивные цифровые следы, как отправленные сообщения, содержание понравившихся постов, количество друзей, количество публикаций, количество фотографий и количество отправленных сообщений. В результате анализа перечисленных следов с помощью экспертной системы на основе нечеткого анализа удалось получить прогноз профориентации человека.

В результате выполнения был сделан вывод о том, что система довольно легко модифицируется: система разделена на модули, взаимодействующие между собой. Поэтому для более точного прогноза в дальнейшем можно добавить модули распознавания жанра музыки, распознавания темы видео, распознавания контента на фотографии и т.д. Так же улучшения качества прогнозирования системы можно добиться, опросив большее количество экспертов, и чем больше направлений будут представлять эксперты, тем качественнее окажется ответ экспертной системы.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1 Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. – СПб.: Питер, 2018. – 480 с.

2 Baccianella S. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining, 2010. – P. 2200-2204.

3 Толстиков, И. В. Прогнозирование профориентации человека по его цифровым следам методами Data Mining / И. В. Толстиков, Т. А. Приходько. Сборник «Прикладная математика: современные проблемы математики, информатики и моделирования». Материалы всероссийской научно-практической конференции. КубГУ. - 2022, выпуск 4. — Текст : непосредственный.

4 Rong X. word2vec Parameter Learning Explained [Электронный ресурс]. – URL: https://arxiv.org/pdf/1411.2738.pdf (дата обращения: 22.04.2020).

5 Пазельская А., Соловьев А. Метод определения эмоций в текстах на русском языке. – М.: Ай-Теко, 2011. – С. 510-522.

6 Клековкина М.В., Котельников Е.В. Метод автоматической классификации текстов по тональности, основанный на словаре эмоциональной лексики. – М.: Ай-Теко, 2012. – 6 с.

7 Джеймс Г., Уиттон Д., Хасти Т., Тибширани Р. Введение в статическое обучение с примерами на языке R/ пер. с англ. Мастицкий С. Э. – М.: ДМК Пресс, 2017. – 456 с.

8 Кун М., Джонсон К. Предиктивное моделирование на практике/ пер. с англ. «Питер». – СПб.: Питер, 2019. – 640 с.

9 Миронов А.М. Машинное обучение [Электронный ресурс]. – URL: http://intsys.msu.ru/staff/mironov/machine\_learning\_vol1.pdf (дата обращения 02.10.2020).

10 Саати Т. Принятие решений Метод анализа иерархий/ пер. с англ. Вачнадзе Р.Г. – М.: Радио и связь, 1993. – 278 с.

11 Климов Е.А. Классификация профессий [Электронный ресурс]. – URL: <https://www.cposo.ru/images/docs/Klassifikacija_professij_E.A.Klimova.pdf> (Дата обращения 14.04.2022).